Automated Reasoning

A Peek Beyond the Basics

Antti-Juhani Kaijanaho

antkaij@mit.jyu.fi

University of Jyväskylä

Department of Mathematical Information Technology

Overview 1

The course has thus far only considered the very basics of the discipline — basic algorithm schemas and the basic theory.

The problem: *inefficiency!*

We need to avoid combinatorial explosion and infinite branching.

Overview 2

First-order logic is just the beginning. How about

- inductive reasoning?
- second-order logic?
- higher-order logic?
- modal logics?
- many-valued logics?

Combinatorial explosion and infinite branching 1

What causes combinatorial explosion or infinite branching?

- the equality symbol
- poor normal forms
- indiscriminating choices
- $P \stackrel{?}{=} NP$
- undecidability

Combinatorial explosion and infinite branching 2

Almost all theories include equality. Equality rules allow generating lots of pointless facts. For example

$$\frac{e = e'}{f(e) = f(e')}$$

generates

$$a = b \Rightarrow f(a) = f(b) \Rightarrow f(f(a)) = f(f(b)) \Rightarrow \ldots$$

All conjunctive normal forms are equal but some are more equal than others. Compare:

$$p \rightarrow ((q \vee p) \rightarrow r) \Rightarrow \neg p \vee \neg(q \vee p) \vee r$$
$$\Rightarrow \neg p \vee (\neg q \wedge \neg p) \vee r$$
$$\Rightarrow (\neg p \vee \neg q \vee r) \wedge (\neg p \vee \neg p \vee r)$$

$$p \rightarrow ((q \vee p) \rightarrow r) \Rightarrow p \rightarrow r$$
$$\Rightarrow \neg p \vee r$$

Combinatorial explosion and infinite branching 4

Mindless decision-making leads to a senseless waste of resources.

Completing your proof program is only the beginning: most of the work is in tuning its heuristics and strategies.

Combinatorial explosion and infinite branching 5

- SAT is NP-complete.
- Sentential validity is co-NP.

$$P = NP$$

- $$\frac{NP = \text{co-}NP}{\text{validity} \in P}$$

That would be a start.

Combinatorial explosion and infinite branching 6

First-order validity is undecidable.

Unfortunately there is nothing we can do about it.

Combinatorial explosion and infinite branching 7

- The equality symbol
- Poor normal forms
- Indiscriminating choices
- $P \stackrel{?}{=} NP$
- Undecidability

cause combinatorial explosion and infinite branching.

Standard equality axioms

$$(\forall x)(x = x)$$

$$(\forall x)(\forall y)(x = y \rightarrow f(x) = f(y))$$

$$(\forall x)(\forall y)(x = y \rightarrow (r(x) \rightarrow r(y)))$$

are unusable for automation due to infinite branching.

Free-variable tableaux equality replacement rule:

If, on a branch, one can find an equality literal T $t = u$, and a literal $\ell(t)$, then the branch may be expanded with $\ell(u)$.[a]

[a] $\ell(t)$ denotes a literal containing t as a subterm and $\ell(u)$ denotes the same literal after every occurrence of t has been replaced by u.

Free-variable tableaux reflexivity rule:

One is allowed at any time expand any branch with the literal T $x = x$, where x is any free variable.

Free-variable tableaux function reflexivity rule:

One is allowed to at any time expand any branch with the literal

$$T\ f(x_1, \ldots, x_n) = f(x_1, \ldots, x_n),$$

where f *is a function symbol of arity* n *and* x_i *are free variables.*

Recall the equality replacement rule:

If, on a branch, one can find an equality literal T $t = u$, and a literal $\ell(t)$, then the branch may be expanded with $\ell(u)$.

A more useful rule is obtained by combining this with substitutions.

Free-variable tableaux equality MGU-replacement rule:

If, on a branch, one can find an equality literal T $t = u$, and a literal $\ell(t')$ such that $\mathrm{mgu}(t, t')$ exists, then the branch may be expanded with $\ell(u)$, provided that $\mathrm{mgu}(t, t')$ is then applied to the whole tableau.

Free-variable tableaux with equality 6

Now, try proving

$$(\forall x)(\exists y)(y = f(x) \wedge (\forall z)(z = f(x) \rightarrow y = z))$$

using free-variable tableaux with equality.

Equality 2

Other equality handling techniques:

- paramodulation
- Knuth–Bendix procedure
- equality elimination

Use them all, not just one!

Recall Overview 2

First-order logic is just the beginning. How about

- inductive reasoning?
- second-order logic?
- higher-order logic?
- modal logics?
- many-valued logics?

Second-order logic 1

Extend first-order logic by assuming a second set of variables.

- Lower-case (first-order) variables $x, y, z, \ldots$ denote individuals.
- Upper-case (second-order) variables $X, Y, Z, \ldots$ denote relations over individuals.

Second-order logic 2

- Make sure that arities of second-order variables are unambiguous in a formula.
- Allow quantification of second-order variables:

$$(\forall X)($$
$$(X(0) \wedge (\forall x)(X(n) \rightarrow X(n+1)))$$
$$\rightarrow (\forall x)(X(x)))$$

Second-order logic 3

- Most interesting mathematical theories are categorical in second-order logic.
- The price: even second-order unification is undecidable.

Higher-order logic 1

- In higher-order logic (HOL), there is no distinction between formulae and terms.
- Terms have types; formulae are simply terms with type Bool.
- Based on Church's simplified type theory and λ-calculus.

Higher-order logic 2

HOL terms are

- variables $x, y, z, \ldots$
- constants $\top, \bot, (\wedge), \forall, \exists, c, \ldots$
- function abstractions $(\lambda x)(t)$
- function applications $t\ u$

HOL terms are typed as follows:

$x : \alpha$	variables are polymorphic
$c : \tau$	constants have associated types
$t\ u : \tau_2$	if $t : \tau_1 \to \tau_2$ and $u : \tau_1$
$(\lambda x)(t) : \tau_1 \to \tau_2$	if $t : \tau_2$ when $x : \tau_1$

For example

$$(\lambda x)(\top) : \alpha \to \mathsf{Bool}$$

$$(\wedge) : \mathsf{Bool} \to \mathsf{Bool} \to \mathsf{Bool}$$

$$\forall : (\alpha \to \mathsf{Bool}) \to \mathsf{Bool}$$

$$\exists : (\alpha \to \mathsf{Bool}) \to \mathsf{Bool}$$

Higher-order logic 4

- HOL is a popular *metalogic*.
- Higher-order unification is undecidable.
- HOL has no complete proof procedure.

Modal logics 1

Modal languages extend sentential and first-order languages with modal operators, for example:

- $\Diamond p$: p *is possibly true*
- $\Box p$: p *is necessarily true*

(How would you read $\Diamond\Box p$?)

Propositional dynamic logic (PDL) is an important modal language:

> *For every nondeterministic program* π *and every well-formed formula* p, $\langle\pi\rangle p$ *and* $[\pi]p$ *are well-formed formulae.*

Additionally, every sentential formula is a wff.

Informal PDL semantics:

- $\langle \pi \rangle p$: any angelic execution of π started now makes p true
- $[\pi] p$: any demonic execution of π started now makes p true

PDL is interesting because it can handle structured programs:

$[\pi_1;\pi_2]p \leftrightarrow [\pi_1][\pi_2]p$	sequencing
$[\pi_1 \cup \pi_2]p \leftrightarrow [\pi_1]p \wedge [\pi_2]p$	nondeterministic choice
$[\pi^*]p \rightarrow p \wedge [\pi^*]p$	nondeterministic repetition
$[q?]p \leftrightarrow q \rightarrow p$	test

For example, $(p?;\pi_1) \cup ((\neg p)?;\pi_2)$ is an encoding of `if` p `then` π_1 `else` π_2.

Modal logics 5

- Modal logics have many more applications in computer science.
- Propositional modal logics are stronger than sentential logic and usually weaker than first-order logic.
- Propositional modal logics are decidable.

Modal tableaux 1

- Modal tableaux are an extension of sentential tableaux.
- Tableau node labels are now *labelled formulae*, that is, of the form $n : p$, where n is a positive integer and p is a formula.

Modal tableaux 2

- $R(n, m)$, where n and m are positive integers, is also a valid node label.
- When a proof of p is sought, the root is labelled with $1 : q$, where q is the negation normal form of $\neg p$.
- Sentential tableau extension rules just copy the n.

Diamond expansion rule:

If, on a branch, a node labelled with $n : \Diamond p$ is found, then the leaf node of that branch may be given a child labelled $R(n, m)$ which itself has a child labelled $m : p$, where m is new to the tableau.

Box expansion rule:

If, on a branch, a node labelled with $n : \Box p$ *and a node labelled with* $R(n, m)$ *is found, then the leaf node of that branch may be given a child labelled* $m : p$.

Modal tableaux 5

- A branch is closed if there are two nodes on it labelled $n : p$ and $n : \neg p$ for some atomic formula p and some positive integer n.
- A tableau is closed if all its branches are closed.

Perhaps you'd like to try it on

$$\Box(p \to q) \to (\Box p \to \Box q)?$$

Conclusion 1

- second-order logic!
- higher-order logic!
- modal logics!

. . . and many more.

First-order logic is just the beginning.

Conclusion 2

We need to avoid combinatorial explosion and infinite branching.

Thus we get: *efficiency!*

Remember to tune your programs!

Thank you

The automated reasoning discipline is a vast one. We just scratched the surface.

Free advertisement: There are dissertation (master's and above) topics buried here. Are you interested?

Qapla'